Privacy-preserving Collaborative Data Mining

Zhijun Zhan[1] and LiWu Chang[2]
[1]Department of Electrical Engineering and Computer Science, Syracuse University, USA
[2]Center for High Assurance Computer Systems, Naval Research Laboratory, USA
Email: zhzhan@ecs.syr.edu and lchang@itd.nrl.navy.mil

Abstract

Privacy-preserving data mining is an important issue in the areas of data mining and security. In this paper, we study how to conduct association rule mining, one of the core data mining techniques, on private data in the following scenario: Multiple parties, each having a private data set, want to jointly conduct association rule mining without disclosing their private data to other parties. Because of the interactive nature among parties, developing a secure framework to achieve such a computation is both challenging and desirable. In this paper, we present a secure framework for multiple parties to conduct privacy-preserving association rule mining.

Key Words: *privacy, security, association rule mining, secure multi-party computation.*

1 INTRODUCTION

Business successes are no longer the result of an individual toiling in isolation; rather successes are dependent upon collaboration, team efforts, and partnership. In the modern business world, collaboration becomes especially important because of the mutual benefit it brings. Sometimes, such a collaboration even occurs among competitors, or among companies that have conflict of interests, but the collaborators are aware that the benefit brought by such a collaboration will give them an advantage over other competitors. For this kind of collaboration, data's privacy becomes extremely important: all the parties of the collaboration promise to provide their private data to the collaboration, but neither of them wants each other or any third party to learn much about their private data.

This paper studies a very specific collaboration that becomes more and more prevalent in the business world. The problem is the collaborative data mining. Data mining is a technology that emerges as a means for identifying patterns and trends from a large quantity of data. The goal of our studies is to develop technologies to enable multiple parties to conduct data mining collaboratively without disclosing their private data.

In recent times, the explosion in the availability of various kinds of data has triggered tremendous opportunities for collaboration, in particular collaboration in data mining. The following is some realistic scenarios:

1. Multiple competing supermarkets, each having an extra large set of data records of its customers' buying behaviors, want to conduct data mining on their joint data set for mutual benefit. Since these companies are competitors in the market, they do not want to disclose too much about their customers' information to each other, but they know the results obtained from this collaboration could bring them an advantage over other competitors.

2. Several pharmaceutical companies, each have invested a significant amount of money conducting experiments related to human genes with the goal of discovering meaningful patterns among the genes. To reduce the cost, the companies decide to join force, but neither wants to disclose too much information about their raw data because they are only interested in this collaboration; by disclosing the raw data, a company essentially enables other parties to make discoveries that the company does not want to share with others.

To use the existing data mining algorithms, all parties need to send their data to a trusted central place (such as a super-computing center) to conduct the mining. However, in situations with privacy concerns, the parties may not trust anyone. We call this type of problem the *Privacy-preserving Collaborative Data Mining* (PCDM) problem. For each data min-

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2003		**00-00-2003 to 00-00-2003**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Privacy-preserving Collaborative Data Mining	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Research Laboratory,Center for High Assurance Computer Systems,4555 Overlook Avenue, SW,Washington,DC,20375**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**8**	

ing problem, there is a corresponding PCDM problem. Fig.1 shows how a traditional data mining problem could be transformed to a PCDM problem (this paper only focuses on the heterogeneous collaboration (Fig.1.c))(heterogeneous collaboration means that each party has different sets of attributes. Homogeneous collaboration means that each party has the same sets of attributes.)

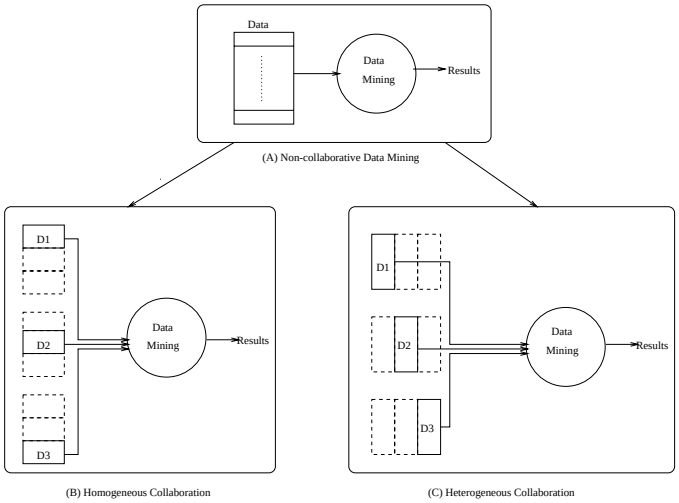

Figure 1. Privacy Preserving Non-collaborative and Collaborative Data Mining

Generic solutions for any kind of secure collaborative computing exist in the literature [5]. These solutions are the results of the studies of the Secure Multi-party Computation problem [10, 5], which is a more general form of secure collaborative computing. However, none of the proposed generic solutions is practical; they are not scalable and cannot handle large-scale data sets because of the prohibitive extra cost in protecting data's privacy. Therefore, practical solutions need to be developed. This need underlies the rationale for our research.

Data mining includes a number of different tasks, such as association rule mining, classification, and clustering. This paper studies the association rule mining problem. The goal of association rule mining is to discover meaningful association rules among the attributes of a large quantity of data. For example, let us consider the database of a medical study, with each attribute representing a symptom found in a patient. A discovered association rule pattern could be "70% of patients who are drug injection takers also have hepatitis". This information can be useful for the disease-control, medical research, etc. Based on the existing association rule mining technologies, we study the *Mining Association Rules On Private Data* (MAP) problem defined as follows: multiple parties want to conduct association rule mining on a data set that consist all the parties' private data, and neither party is willing to disclose its raw data to other parties.

The existing research on association rule mining [8] provides the basis for the collaborative association rule mining. However, none of those methods satisfy the security requirements of MAP or can be trivially modified to satisfy them. With the increasing needs of privacy-preserving data mining, more and more people are interested in finding solutions to the MAP problem. Vaidya and Clifton proposed a solution [9] for two parties to conduct privacy-preserving association rule mining. However, for the general case where more than two parties are involved, the MAP problem presents a much greater challenge.

The paper is organized as follows: The related work is discussed in Section 2. We describe the association rule mining procedure in Section 3. We then formally define our proposed secure protocol in Section 4. In Section 5, we conduct security and communication analysis. We give our conclusion in Section 6.

2 RELATED WORK

Secure Multi-party Computation

Briefly, a Secure Multi-party Computation (SMC) problem deals with computing any function on any input, in a distributed network where each participant holds one of the inputs, while ensuring that no more information is revealed to a participant in the computation than can be inferred from that participant's input and output. The SMC problem literature was introduced by Yao [10]. It has been proved that for any function, there is a secure multi-party computation solution [5]. The approach used is as follows: the function F to be computed is first represented as a combinatorial circuit, and then the parties run a short protocol for every gate in the circuit. Every participant gets corresponding shares of the input wires and the output wires for every gate. This approach, although appealing in its generality and simplicity, is highly impractical.

Privacy-preservation Data Mining

In the early work on such a privacy-preserving data mining problem, Lindell and Pinkas [7] propose a solution to the privacy-preserving classification problem using the oblivious transfer protocol, a powerful tool developed through the secure multi-party computation studies. Another approach for solving the privacy-preserving classification problem was proposed by Agrawal and Srikant [1]. In their approach, each individual data item is perturbed and the distributions of

the all data is reconstructed at an aggregate level. The technique works for those data mining algorithms that use the probability distributions rather than individual records. In [9], a solution to the association mining problem for the case of two parties was proposed. In [3], a procedure is provided to build a classifier on private data, where a semi-trusted party was employed to improve the performance of communication and computation. In this paper, we also adopt the model of the semi-trusted party because of the effectiveness and usefulness it brings and present a secure protocol allowing computation to be carried out by the parties.

3 MINING ASSOCIATION RULES ON PRIVATE DATA

Since its introduction in 1993 [8], the association rule mining has received a great deal of attention. It is still one of most popular pattern-discovery methods in the field of knowledge discovery. Briefly, an association rule is an expression $X \Rightarrow Y$, where X and Y are sets of items. The meaning of such rules is as follows: Given a database D of records, $X \Rightarrow Y$ means that whenever a record R contains X then R also contains Y with certain confidence. The rule confidence is defined as the percentage of records containing both X and Y with regard to the overall number of records containing X. The fraction of records R supporting an item X with respect to database D is called the support of X.

3.1 Problem Definition

We consider the scenario where multiple parties, each having a private data set (denoted by D_1, D_2, $\cdots$, and D_n respectively), want to collaboratively conduct association rule mining on the union of their data sets. Because they are concerned about their data's privacy, neither party is willing to disclose its raw data set to others. Without loss of generality, we make the following assumptions on the data sets. The assumptions can be achieved by pre-processing the data sets D_1, D_2, $\cdots$, and D_n, and such a pre-processing does not require one party to send its data set to other parties. (In this paper, we consider applications where the identifier of each data record is recorded. In contrast, for transactions such as the supermarket-buying, customers' IDs may not be needed. The IDs and the names of attributes are known to all parties during the joint computation. A data record used in the joint association rule mining has the same ID in different databases.)

1. D_1, D_2, $\cdots$ and D_n are binary data sets, namely they only contain 0's and 1's, where n is the total number of parties. (Our method is applicable to attributes that are of non-binary value. An attribute of non-binary value will be converted to a binary representation. Detailed implementation includes discretizing and categorizing attributes that are of continuous or ordinal values.)

2. D_1, D_2, $\cdots$ and D_n contain the same number of records. Let N denote the total number of records for each data set.

3. The identities of the ith (for $i \in [1, N]$) record in D_1, D_2, $\cdots$ and D_n are the same.

Mining Association Rules On Private Data problem: Party 1 has a private data set D_1, party 2 has a private data set D_2, $\cdots$ and party n has a private data set D_n. The data set $[D_1 \cup D_2 \cup \cdots \cup D_n]$ is the union of D_1, D_2, $\cdots$ and D_n (by vertically putting D_1, D_2, $\cdots$ and D_n together so that the concatenation of the ith row in D_1, D_2, $\cdots$ and D_n becomes the ith row in $[D_1 \cup D_2 \cup \cdots \cup D_n]$). The n parties want to conduct association rule mining on $[D_1 \cup D_2 \cup \cdots \cup D_n]$ and to find the association rules with support and confidence being greater than the given thresholds. We say an association rule (e.g., $x_i \Rightarrow y_j$) has confidence $c\%$ in the data set $[D_1 \cup D_2 \cup \cdots \cup D_n]$ if in $[D_1 \cup D_2 \cup \cdots \cup D_n]$ $c\%$ of the records which contain x_i also contain y_j (namely, $c\% = P(y_j \mid x_i)$). We say that the association rule has support $s\%$ in $[D_1 \cup D_2 \cup \cdots \cup D_n]$ if $s\%$ of the records in $[D_1 \cup D_2 \cdots \cup D_n]$ contain both x_i and y_j (namely, $s\% = P(x_i \cap y_j)$).

3.2 Association Rule Mining Procedure

The following is the procedure for mining association rules on $[D_1 \cup D_2 \cdots \cup D_n]$.

1. L_1 = large 1-itemsets
2. **for** (k = 2; $L_{k-1} \neq \phi$; k++) **do begin**
3. C_k = **apriori-gen**(L_{k-1})
4. **for** all candidates $c \in C_k$ **do begin**
5. **Compute *c.count*** \\ We will show how to compute it in Section 3.3
6. **end**
7. $L_k = \{c \in C_k | c.count \geq min\text{-}sup\}$
8. **end**
9. Return L = $\cup_k L_k$

The procedure **apriori-gen** is described in the following (please also see [6] for details).

apriori-gen(L_{k-1}: large (k-1)-itemsets)

1. **for** each itemset $l_1 \in L_{k-1}$ **do begin**

2. **for** each itemset $l_2 \in L_{k-1}$ **do begin**
3. **if** $((l_1[1] = l_2[1]) \wedge (l_1[2] = l_2[2]) \wedge \cdots \wedge (l_1[k-1] = l_2[k-1]) \wedge (l_1[k-1] < l_2[k-1]))\{$
4. **then** c = l_1 *join* l_2
5. **for each** (k-1)-subset s of c **do begin**
6. **if** s $\notin L_{k-1}$
7. **then** delete c
8. **else** add c to C_k
9. **end**
10. }
11. **end**
12. **end**
13. return C_k

3.3 How to compute *c.count*

If all the candidates belong to the same party, then *c.count*, which refers to the frequency counts for candidates, can be computed by this party. If the candidates belong to different parties, they then construct vectors for their own attributes and apply our number product protocol, which will be discussed in Section 4, to obtain the *c.count*. We use an example to illustrate how to compute *c.count* among three parties. Party 1, party 2 and party 3 construct vectors X, Y and Z for their own attributes respectively. To obtain *c.count*, they need to compute $\sum_{i=1}^{N} X[i] \cdot Y[i] \cdot Z[i]$ where N is the total number of values in each vector. For instance, if the vectors are as depicted in Fig.2, then $\sum_{i=1}^{N} X[i] \cdot Y[i] \cdot Z[i] = \sum_{i=1}^{5} X[i] \cdot Y[i] \cdot Z[i] = 3$. We provide an efficient protocol in Section 4 for the parties to compute this value without revealing their private data to each other.

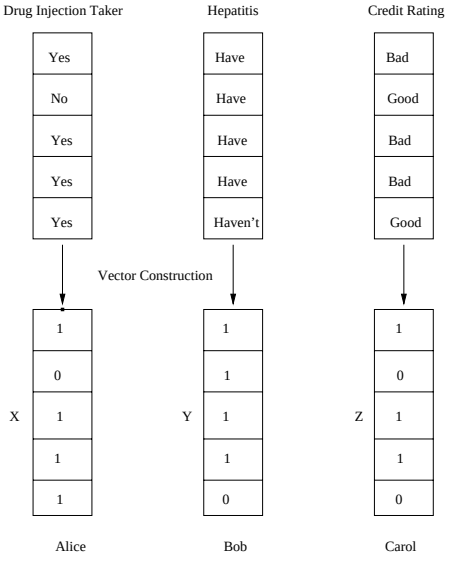

Figure 2. Raw Data For Alice, Bob and Carol

4 BUILDING BLOCK

How two or multiple parties jointly compute *c.count* without revealing their raw data to each other is the challenge that we want to address. The number product protocol described in this section is the main technical tool used to compute it. We will describe an efficient solution of the number product protocol based on a commodity server, a semi-trusted party.

Our building blocks are two protocols: The first protocol is for two parties to conduct the multiplication operation. This protocol differs from [3] in that we consider the product of numbers instead of vectors. Since product computation can only applied for two vectors, it cannot deal with the computation involved in multiple parties where more than two vectors may participate in the computation. The second protocol, with the first protocol as the basis, is designed for the secure multi-party product operation.

4.1 Introducing The Commodity Server

For performance reasons, we use an extra server, the commodity server [2] in our protocol. The parties could send requests to the commodity server and receive data (called *commodities*) from the server, but the commodities must be independent of the parties' private data. The purpose of the commodities is to help the parties conduct the desired computations.

The commodity server is semi-trusted in the following senses: (1) It should not be trusted; therefore it should not be possible to derive the private information of the data from the parties; it should not learn the computation result either. (2) It should not collude with all the parties. (3) It follows the protocol correctly. Because of these characteristics, we say that it is a semi-trusted party. In the real world, finding such a semi-trusted party is much easier than finding a trusted party.

As we will see from our solutions, the commodity server does not participate in the actual computation among the parties; it only supplies commodities that are independent of the parties' private data. Therefore, the server can generate independent data off-line beforehand, and sell them as commodities to the prover and the verifier (hence the name "commodity server").

4.2 Secure Number Product Protocol

Let's first consider the case of two parties where $n = 2$ (more general cases where $n \geq 3$ will be discussed later). Alice has a vector X and Bob has a vector Y. Both vectors have N elements. Alice and Bob want

to compute the product between X and Y such that Alice gets $\sum_{i=1}^{N} U_x[i]$ and Bob gets $\sum_{i=1}^{N} U_y[i]$, where $\sum_{i=1}^{N} U_x[i] + \sum_{i=1}^{N} U_y[i] = \sum_{i=1}^{N} X[i] \cdot Y[i] = X \cdot Y$. $U_y[i]$ and $U_x[i]$ are random numbers. Namely, the scalar product of X and Y is divided into two secret pieces, with one piece going to Alice and the other going to Bob. We assume that random numbers are generated from the integer domain.

4.2.1 Secure Two-party Product Protocol

Protocol 1. *(Secure Two-party Product Protocol)*

1. The Commodity Server generates two random numbers $R_x[1]$ and $R_y[1]$, and lets $r_x[1] + r_y[1] = R_x[1] \cdot R_y[1]$, where $r_x[1]$ (or $r_y[1]$) is a randomly generated number. Then the server sends $(R_x[1], r_x[1])$ to Alice, and $(R_y[1], r_y[1])$ to Bob.
2. Alice sends $\hat{X}[1] = X[1] + R_x[1]$ to Bob.
3. Bob sends $\hat{Y}[1] = Y[1] + R_y[1]$ to Alice.
4. Bob generates a random number $U_y[1]$, and computes $\hat{X}[1] \cdot Y[1] + (r_y[1] - U_y[1])$, then sends the result to Alice.
5. Alice computes $(\hat{X}[1] \cdot Y[1] + (r_y[1] - U_y[1])) - (R_x[1] \cdot \hat{Y}[1]) + r_x[1] = X[1] \cdot Y[1] - U_y[1] + (r_y[1] - R_x[1] \cdot R_y[1] + r_x[1]) = X[1] \cdot Y[1] - U_y[1] = U_x[1]$.
6. Repeat step 1-5 to compute $X[i] \cdot Y[i]$ for $i \in [2, N]$. Alice then gets $\sum_{i=1}^{N} U_x[i]$ and Bob gets $\sum_{i=1}^{N} U_y[i]$.

The bit-wise communication cost of this protocol is $7 * M * N$, where M is the maximum bits for the values involved in our protocol. The cost is approximately 7 times of the *optimal* cost of a two-party scalar product (the optimal cost of a scalar product is defined as the cost of conducting the product of X and Y without the privacy constraints, namely one party simply sends its data in plain to the other party). The cost can be decreased to $3 * M * N$ if the commodity server just sends seeds to Alice and Bob since the seeds can be used to generate a set of random numbers.

4.2.2 Secure Multi-party Product Protocol

We have discussed our protocol of secure number product for two parties. Next, we will consider the protocol for securely computing the number product for multiple parties. For simplicity, we only describe the protocol when $n = 3$. The protocols for the cases when $n > 3$ can be similarly derived. Our concern is that similarly derived solution may not efficient when the number of parties is large, and more efficient solution is still under research. Without loss of generality, let Alice has a private vector X and a randomly generated vector R_x, Bob has a private vector Y and a randomly generated vector R_y and let $R_y[i] = R'_y[i] + R''_y[i]$ for $i \in [1, N]$ and Carol has a private vector Z and a randomly generated vector R_z. First, we let the parties hide these private numbers by using their respective random numbers, then conduct the product for the multiple numbers.

$$
\begin{aligned}
&T[1] = \\
&(X[1] + R_x[1]) * (Y[1] + R_y[1]) * (Z[1] + R_z[1]) \\
&\\
&= X[1]Y[1]Z[1] + X[1]R_y[1]Z[1] + R_x[1]Y[1]Z[1] \\
&+R_x[1]R_y[1]Z[1] + X[1]Y[1]R_z[1] + X[1]R_y[1]R_z[1] \\
&+R_x[1]Y[1]R_z[1] + R_x[1]R_y[1]R_z[1] \\
&\\
&= \\
&X[1]Y[1]Z[1] + X[1]R_y[1]Z[1] + R_x[1]Y[1]Z[1] \\
&+R_x[1]R_y[1]Z[1] + X[1]Y[1]R_z[1] + X[1]R_y[1]R_z[1] \\
&+R_x[1]Y[1]R_z[1] + R_x[1](R'_y[1] + R''_y[1])R_z[1] \\
&\\
&= X[1]Y[1]Z[1] + X[1]R_y[1]Z[1] + R_x[1]Y[1]Z[1] \\
&+R_x[1]R_y[1]Z[1] + X[1]Y[1]R_z[1] + X[1]R_y[1]R_z[1] \\
&+R_x[1]Y[1]R_z[1] + R_x[1]R'_y[1]R_z[1] + R_x[1]R''_y[1]R_z[1] \\
&\\
&= \\
&X[1]Y[1]Z[1] + (X[1]R_y[1] + R_x[1]Y[1] + R_x[1]R_y[1])Z[1] \\
&+(X[1]Y[1] + X[1]R_y[1] + R_x[1]Y[1] + R_x[1]R'_y[1])R_z[1] \\
&+R_x[1]R''_y[1]R_z[1] = T_0[1] + T_1[1] + T_2[1] + T_3[1],
\end{aligned}
$$

where

$$
\begin{aligned}
&T_0[1] = X[1]Y[1]Z[1], \\
&T_1[1] = (X[1]R_y[1] + R_x[1]Y[1] + R_x[1]R_y[1])Z[1], \\
&T_2[1] \\
&= (X[1]Y[1] + X[1]R_y[1] + R_x[1]Y[1] + R_x[1]R'_y[1])R_z[1], \\
&T_3[1] = R_x[1]R''_y[1]R_z[1].
\end{aligned}
$$

$T_0[1]$ is what we want to obtain. To compute $T_0[1]$, we need to know $T[1]$, $T_1[1]$, $T_2[1]$ and $T_3[1]$. In this protocol, we let Alice get $T[1]$, Bob get $T_3[1]$ and Carol get $T_1[1]$ and $T_2[1]$. Bob separates $R_y[1]$ into $R'_y[1]$ and $R''_y[1]$. If he fails to do so, then his data might be disclosed during the computation of these terms.

To compute $T_1[1]$ and $T_2[1]$, Alice and Bob can use Protocol 1 to compute $X[1]R_y[1]$, $R_x[1]Y[1]$, $R_x[1]R_y[1]$, $X[1]Y[1]$ and $R_x[1]R'_y[1]$. Thus, according to Protocol 1, Alice gets $U_x[1]$, $U_x[2]$, $U_x[3]$, $U_x[4]$ and $U_x[5]$ and Bob gets $U_y[1]$, $U_y[2]$, $U_y[3]$, $U_y[4]$ and $U_y[5]$. Then they compute $(X[1]R_y[1] + R_x[1]Y[1] + R_x[1]R_y[1])$ and $(X[1]Y[1] + X[1]R_y[1] + R_x[1]Y[1] + R_x[1]R'_y[1])$ and send the results to Carol who can then compute $T_1[1]$ and $T_2[1]$. Note that $X[1]R_y[1] = U_x[1] + U_y[1]$, $R_x[1]Y[1] = U_x[2] + U_y[2]$,

$R_x[1]R_y[1] = U_x[3] + U_y[3]$, $X[1]Y[1] = U_x[4] + U_y[4]$ and $R_x[1]R'_y[1] = U_x[5] + U_y[5]$.

To compute $T_3[1]$, Alice and Carol use Protocol 1 to compute $R_x[1]R_z[1]$, then send the results to Bob who can then compute $T_3[1]$.

To compute T[1], Bob sends $Y[1] + R_y[1]$ to Alice and Carol sends $Z[1] + R_z[1]$ to Alice.

Repeat the above process to compute $T[i]$, $T_1[i]$, $T_2[i]$, $T_3[i]$ and $T_0[i]$ for $i \in [2, N]$. Then, Alice has $\sum_{i=1}^{N} T[i]$, Bob has $\sum_{i=1}^{N} T_3[i]$ and Carol has $\sum_{i=1}^{N} T_1[i]$ and $\sum_{i=1}^{N} T_2[i]$. Finally, we achieve the goal and obtain $\sum_{i=1}^{N} X[i]Y[i]Z[i] = \sum_{i=1}^{N} T_0[i] = \sum_{i=1}^{N} T[i] - \sum_{i=1}^{N} T_1[i] - \sum_{i=1}^{N} T_2[i] - \sum_{i=1}^{N} T_3[i]$.

Protocol 2. *(Secure Multi-party Product Protocol)*
Step I: Random Number Generation

1. Alice generates a random number $R_x[1]$.
2. Bob generates two random numbers $R'_y[1]$ and $R''_y[1]$.
3. Carol generates a random number $R_z[1]$.

Step II: Collaborative Computing
Sub-step 1: To Compute $T[1]$

1. Carol computes $Z[1] + R_z[1]$ and sends it to Bob.
2. Bob computes $Y[1] + R_y[1]$ and sends it to Alice.
3. Alice computes $T[1] = (X[1] + R_x[1]) * (Y[1] + R_y[1]) * (Z[1] + R_z[1])$.

Sub-step 2: To Compute $T_1[1]$ and $T_2[1]$

1. Alice and Bob use Protocol 1 to compute $X[1] \cdot R_y[1]$, $R_x[1] \cdot Y[1]$, $R_x[1] \cdot R_y[1]$, $X[1] \cdot Y[1]$ and $R_x[1] \cdot R'_y[1]$. Then Alice obtains $U_x[1]$, $U_x[2]$, $U_x[3]$, $U_x[4]$ and $U_x[5]$ and Bob obtains $U_y[1]$, $U_y[2]$, $U_y[3]$, $U_y[4]$ and $U_y[5]$.
2. Alice sends $(U_x[1] + U_x[2] + U_x[3])$ and $(U_x[4] + U_x[1] + U_x[2] + U_x[5])$ to Carol.
3. Bob sends $(U_y[1] + U_y[2] + U_y[3])$ and $(U_y[4] + U_y[1] + U_y[2] + U_y[5])$ to Carol.
4. Carol computes
   $T_1[1] = (X[1] \cdot R_y[1] + R_x[1] \cdot Y[1] + R_x[1] \cdot R_y[1]) \cdot Z[1]$, and
   $T_2[1] = (X[1] \cdot Y[1] + X[1] \cdot R_y[1] + R_x[1] \cdot Y[1] + R_x[1] \cdot R'_y[1]) \cdot R_z[1]$.

Sub-step 3: To Compute $T_3[1]$

1. Alice and Carol use Protocol 1 to compute $R_x[1] \cdot R_z[1]$ and send the values they obtained from the protocol to Bob.
2. Bob computes $T_3[1] = R''_y[1] \cdot R_x[1] \cdot R_z[1]$.

Step III: Repeating

1. Repeat the Step I and Step II to compute $T[i]$, $T_1[i]$, $T_2[i]$ and $T_3[i]$ for $i \in [2, N]$.
2. Alice then gets
   $[a] = \sum_{i=1}^{N} T[i] = \sum_{i=1}^{N} (X[i] + R_x[i]) * (Y[i] + R_y[i]) * (Z[i] + R_z[i])$.
3. Bob gets
   $[b] = \sum_{i=1}^{N} T_3[i] = \sum_{i=1}^{N} (Y[i] + R_y[i]) * (Z[i] + R_z[i])$.
4. Carol gets
   $[c] = \sum_{i=1}^{N} T_1[i] = \sum_{i=1}^{N} (X[i] \cdot R_y[i] + R_x[i] \cdot Y[i] + R_x[i] \cdot R_y[i]) \cdot Z[i]$, and
   $[d] = \sum_{i=1}^{N} T_2[i] = \sum_{i=1}^{N} (X[i] \cdot Y[i] + X[i] \cdot R_y[i] + R_x[i] \cdot Y[i] + R_x[i] \cdot R'_y[i]) \cdot R_z[i]$.
   Note that $\sum_{i=1}^{N} X[i] \cdot Y[i] \cdot Z[i] = \sum_{i=1}^{N} T_0[i] = [a] - [b] - [c] - [d]$.

Theorem 1. *Protocol 1 is secure such that Alice cannot learn Y and Bob cannot learn X either.*

Proof. The number $\hat{X}[i] = X[i] + R_x[i]$ is all what Bob gets. Because of the randomness and the secrecy of $R_x[i]$, Bob cannot find out $X[i]$. According to the protocol, Alice gets (1) $\hat{Y}[i] = Y[i] + R_y[i]$, (2) $Z[i] = \hat{X}[i] \cdot Y[i] + (r_y[i] - U_y[i])$, and (3) $r_x[i]$, $R_x[i]$, where $r_x[i] + r_y[i] = R_x[i] \cdot R_y[i]$. We will show that for any arbitrary $Y'[i]$, there exists $r'_y[i]$, $R'_y[i]$ and $U'_y[i]$ that satisfies the above equations. Assume $Y'[i]$ is an arbitrary number. Let $R'_y[i] = \hat{Y}[i] - Y'[i]$, $r'_y[i] = R_x[i] \cdot R_y[i] - r_x[i]$, and $U'_y[i] = \hat{X}[i] \cdot Y'[i] + r'_y[i]$. Therefore, Alice has (1) $\hat{Y}[i] = Y'[i] + R'_y[i]$, (2) $Z[i] = \hat{X}[i] \cdot Y'[i] + (r'_y[i] - U'_y[i])$ and (3) $r_x[i]$, $R_x[i]$, where $r_x[i] + r'_y[i] = R_x[i] \cdot R'_y[i]$. Thus, from what Alice learns, there exists infinite possible values for $Y[i]$. Therefore, Alice cannot know Y and neither can Bob know X. □

Theorem 2. *Protocol 2 is secure such that Alice cannot learn Y and Z, Bob cannot learn X and Z, and Carol cannot learn X and Y.*

Proof. According to the protocol, Alice obtains
(1) $(Y[i] + R_y[i])$, and (2)$(Z[i] + R_z[i])$.
Bob gets $R_x[i] \cdot R_z[i]$.
Carol gets
(1) $(X[i] \cdot R_y[i] + R_x[i] \cdot Y[i] + R_x[i] \cdot R_y[i])$ and

(2) $(X[i] \cdot Y[i] + X[i] \cdot R_y[i] + R_x[i] \cdot Y[i] + R_x[i] \cdot R'_y[i])$.

Since $R_x[i]$, $R_y[i](= (R'_y[i] + R''_y[i]))$ and $R_z[i]$ are arbitrary random numbers. From what Alice learns, there exists infinite possible values for $Y[i]$ and $Z[i]$. From what Bob learns, there also exists infinite possible values for $Z[i]$. From what Carol learns, there still exists infinite possible values for $X[i]$ and $Y[i]$.

Therefore, Alice cannot learn Y and Z, Bob cannot learn X and Z, and Carol cannot learn X and Y either. □

5 ANALYSIS

5.1 Security analysis

5.1.1 Why Choose A Large Domain

In our protocols, all the random numbers are generated from a very large domain (e.g., the integer domain). If the random numbers are generated from a small domain, then one party might get some information about the other parties' private data. For example, in the Protocol 1, if the elements of $Y[i]$ are in the domain of [0, 300], and we also know the random numbers are generated from [0, 500], then if an element of the vector $Y[i] + R_y[i]$ is 650, we know the original element in the vector Y_i is larger than 150.

5.1.2 Malicious Model Analysis

In this paper, our algorithm is based on the semi-honest model, where all the parties behave honestly and cooperatively during the protocol execution. However, in practice, one of the parties (e.g., Bob) may be malicious in that it wants to gain true values of other parties' data by purposely manipulating its own data before executing the protocols. For example, in Protocol 1 Bob wants to know whether $X[i] = 1$ for some i. He may make up a set of numbers with all, but the ith, values being set to 0's (i.e., $Y[i] = 1$ and $Y[j] = 0$ for $j \neq i$). According to Protocol 1, if Bob obtains $\sum_{i=1}^{N} U_x[i] + \sum_{i=1}^{N} U_y[i]$, indicating the total number of counts for both X and Y being 1, then Bob can know that $X[i]$ is 0 if the above result is 0 and $X[i]$ is 1 if the above result is 1. To deal with this problem, we may randomly select a party to hold the frequency counts. For example, let's consider the scenario of three parties. Without loss of generality, we assume Bob is a malicious party. The chance that Bob gets chosen to hold the frequency counts is $\frac{1}{3}$. We consider the following two cases.(Assume that the probability of samples in a sample space are equally likely.)

1. Make a correct guess of both Alice's and Carol's values.

   If Bob is not chosen to hold the frequency counts, he then chooses to randomly guess and the probability for him to make a correct guess is $\frac{1}{4}$. In case Bob is chosen, if the product result is 1 (with the probability of $\frac{1}{4}$), he then concludes that both Alice and Carol have value 1; if the product result is 0 (with the probability of $\frac{3}{4}$), he would have a chance of $\frac{1}{3}$ to make a correct guess. Therefore, we have $\frac{2}{3} * \frac{1}{4} + \frac{1}{3}(\frac{1}{4} + \frac{3}{4} * \frac{1}{3}) \approx 33\%$. Note that the chance for Bob to make a correct guess, without his data being purposely manipulated, is 25%.

2. Make a correct guess for only one party's (e.g., Alice) value.

   If Bob is not chosen to hold the frequency counts, the chance that his guess is correct is $\frac{1}{2}$. In case Bob is chosen, if the product result is 1, he then knows the Alice's value with certainty; if the result is 0, there are two possibilities that need to be considered: (1) if Alice's value is 0, then the chance that his guess is correct is $\frac{2}{3}$; (2) if Alice's value is 1, then the chance that his guess is correct is $\frac{1}{3}$. Therefore, we have $\frac{2}{3} * \frac{1}{2} + \frac{1}{3}(\frac{1}{4} + \frac{3}{4}(\frac{2}{3} * \frac{2}{3} + \frac{1}{3} * \frac{1}{3})) \approx 56\%$. However, if Bob chooses to make a random guess, he then has 50% of chance to be correct.

It can be shown that the ratio for Bob to make a correct guess with/without manipulating his data in case 1 is (n+1)/n and in case 2 is approaching 1 with an exponential rate of n ($\approx 2^{-(n-1)}$), where n is the number of parties. The probability for a malicious party to make a correct guess about other parties' values decreases significantly as the number of parties increases.

5.1.3 How to deal with information disclosure by the inference from the results

Assume the association rule, $DrugInjection \Rightarrow Hepatitis$, is what we get from the collaborative association rule mining, and this rule has 99% confidence level (i.e., $P(Hepatitis|DrugInjection) = 0.99$). Now given a data item $item_1$ with Alice:(Drug-Injection), Alice can figure out Bob's data (i.e., Bob:(Hepatitis) is in $item_1$) with confidence 99% (but not vice versa). Such an inference problem exists whenever the items of the association rule is small and its confidence measure is high. To deal with the information disclosure through inference, we may enforce the parties to randomize their data as in [4] with some probabilities before conducting the association rule mining.

5.1.4 How to deal with the repeat use of protocol

A malicious party (e.g., Bob) may ask to run the protocol multiple times with different set of values at each time by manipulating his $U_y[.]$ value. If other parties respond with honest answers, then this malicious party may have chance to obtain actual values of other parties. To avoid this type of disclosure, constraints must be imposed on the number of repetitions.

5.2 Communication Analysis

There are three sources which contribute to the total bit-wise communication cost for the above protocols: (1) the number of rounds of communication to compute the number product for a single value (denoted by $NumRod$); (2) the maximum number of bits for the values involved in the protocols (denoted by M); (2) the number of times (N) that the protocols are applied. The total cost can be expressed by $NumRod * M * N$ where $NumRod$ and M are constants for each protocol. Therefore the communication cost is O(N). N is a large number when the number of parties is big since N exponentially increases as the number of parties expands.

6 CONCLUDING REMARKS

In this paper, we consider the problem of privacy-preserving collaborative data mining with inputs of binary data sets. In particular, we study how multiple parties to jointly conduct association rule mining on private data. We provided an efficient association rule mining procedure to carry out such a computation. In order to securely collecting necessary statistical measures from data of multiple parties, we have developed a secure protocol, namely the number product protocol, for multiple-party to jointly conduct their desired computations. We also discussed the malicious model and approached it by distributing the measure of frequency counts to different parties, and suggested the use of the randomization method to reduce the inference of data disclosure.

In our future work, we will extend our method to deal with non-binary data sets. We will also apply our technique to other data mining computations, such as privacy-preserving clustering.

References

[1] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD on Management of Data*, pages 439–450, Dallas, TX USA, May 15 - 18 2000.

[2] D. Beaver. Commodity-based cryptography (extended abstract). In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, El Paso, TX USA, May 4-6 1997.

[3] W. Du and Z. Zhan. Building decision tree classifier on private data. In *Workshop on Privacy, Security, and Data Mining at The 2002 IEEE International Conference on Data Mining (ICDM'02)*, Maebashi City, Japan, December 9 2002.

[4] W. Du and Z. Zhan. Using randomized response techniques for privacy-preserving data mining. In *Proceedings of The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, August 24-27 2003.

[5] O. Goldreich. Secure multi-party computation (working draft). http://www.wisdom.weizmann.ac.il/home/oded/public_html/foc.html, 1998.

[6] J. Han and M. Kamber. *Data Mining Concepts and Techniques*. Morgan Kaufmann Publishers, 2001.

[7] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in Cryptology - Crypto2000, Lecture Notes in Computer Science*, volume 1880, 2000.

[8] R.Agrawal, T.Imielinski and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of ACM SIGMOD Conference on Management of Data SIGMOD93*, pages 207–216, May 1993.

[9] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July 23-26 2002.

[10] A.C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, 1982.